Name: Levi George

Date: 04/17/2021

Class: CS350 Programming Language Design

Prof. Amal Khalifa

Homework 6: Concurrent Programming

All Work is included in the three files included with this folder. concurrentCode includes all of the code used to program the monte carlo simulation with multiple threads. serialCode includes the code that runs the monte carlo simulation serially. The test.pdf file includes the images from a single run of each test, from 1 – 16 threads and the serial execution of the code.

Additionally, concurrentCode includes my comments on the differences between each run and it’s length of execution. My thoughts on the execution time will also be included here.

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Results of Thread Variation

Average Thread Time Slowest Thread Time

One Thread: 1.3732 ms 1.3732 ms

Two Threads: 1.4802 ms 1.7383 ms

Four Threads: 1.992 ms 2.5967 ms

Eight Threads: 2.538 ms 3.6392 ms

Sixteen Threads: 3.634 ms 6.2366 ms

It appears that increasing the number of threads increases the time required to run

it could be that my parallel programming is not as efficient, so I have each thread stumbling over one another.

However, I believe that it could also be that the size of our test (1000 points) is too small to properly manage and make use of concurrent programming.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/